

Cellular automaton supercolliders

Genaro J. Martínez^{1,2}, Andrew Adamatzky¹
 Christopher R. Stephens², Alejandro F. Hoefflich²

January 22, 2013

¹ Unconventional Computing Centre, University of the West of England,
 Bristol BS16 1QY, United Kingdom.

{genaro.martinez, andrew.adamatzky}@uwe.ac.uk

² Instituto de Ciencias Nucleares and Centro de Ciencias de la Complejidad,
 Universidad Nacional Autónoma de México, México.

{stephens, frank}@nucleares.unam.mx

Abstract

Gliders in one-dimensional cellular automata are compact groups of non-quiescent and non-ether patterns (ether represents a periodic background) translating along automaton lattice. They are cellular-automaton analogous of localizations or quasi-local collective excitations travelling in a spatially extended non-linear medium. They can be considered as binary strings or symbols travelling along a one-dimensional ring, interacting with each other and changing their states, or symbolic values, as a result of interactions. We analyse what types of interaction occur between gliders travelling on a cellular automaton ‘cyclotron’ and build a catalog of the most common reactions. We demonstrate that collisions between gliders emulate the basic types of interaction that occur between localizations in non-linear media: fusion, elastic collision, and soliton-like collision. Computational outcomes of a swarm of gliders circling on a one-dimensional torus are analysed via implementation of cyclic tag systems.

Keywords: cellular automata, particles, travelling localizations, collisions, beam routing, universality

1 Introduction

The era of unconventional computers — implementations of computing schemes in physical, chemical and biological substrates, and interpretation of the behaviour of natural systems’ in terms of computation – has brought us a plethora of original prototypes of future and emerging computing architectures. Some examples of this new paradigm are polymer- and conductive foam based extended analog computers [42], chemical reaction-diffusion computers [21, 4], aromatic

molecular computers [10], slime mould computers [50], micro-fluidic based computers [20], enzymatic logical circuits [25], molecular arithmetical circuits [58] to name but a few. Many of these novel computing devices work on the principle that information is represented by localised states of natural systems (phase or diffusive waves, propagating pseudopodia, electron density, conformation) which can then be represented by cellular automata or other discrete automata models (at least in principle) [1]. Most models of unconventional computers suffer, up to different degrees, from boundary problems. The systems are confined to some experimental arena, e.g. test tube or a Petri dish, or even be controlled and programmed by externally imposed geometrical constraints (e.g. channels in excitable chemical computers). It would be incredible useful to produce a model which is self-contained, is not dependent on external constraints and which can, subject to resources and energy supplied, function indefinitely. A particle “super collider” may be the best candidate for such a universal model of boundary free computation.

In his seminal paper “Symbol super colliders” [60] Tommaso Toffoli envisaged far fetching analogies between physical implementations of particle super colliders, lattice gas and one-dimensional cellular automata. He suggested that the concept of symbol super collider – where myriad of tokens run along intersecting rings and interact with each other to produce new tokens – can be used in designing novel types of nature-inspired computing devices [17, 59]. In present paper we develop Toffoli’s ideas further and provide computational implementations of particle ‘accelerator’ or test rigs, implemented in elementary cellular automata.

The paper is structured as follows. One-dimensional cellular automata, a historic and with memory, are introduced in Sect. 2. The concepts of supercollider and ballistic computing are presented in Sect. 3. Section 4 shows how essential elements of collision-based computers can be implemented via glider interactions in one-dimensional cellular automata, and computational capacities are presented in Sect. 5. Outcomes of the presented results and directions for further studies are discussed in Sect. 6.

2 One-dimensional cellular automata

A cellular automaton (CA) is a quintuple $\langle \Sigma, \varphi, \mu, c_0, L \rangle$ based on a one-dimensional lattice L of cells, where each cell x_i , $i \in N$, takes a state from a finite alphabet Σ . A sequence $s \in \Sigma^n$ of n cell-states represents a string or a global configuration c on Σ . We write a set of finite configurations as Σ^n . Cells update their states via an evolution rule $\varphi : \Sigma^\mu \rightarrow \Sigma$, such that μ represents a cell neighbourhood that consists of a central cell and a number of neighbours connected locally. There are $|\Sigma|^\mu$ different neighbourhoods and if $k = |\Sigma|$ then we have k^{μ} different evolution rules.

An evolution diagram for a CA is represented by a sequence of configurations $\{c_i\}$ generated by the global mapping $\Phi : \Sigma^n \rightarrow \Sigma^n$, where a global relation is given by as $\Phi(c^t) \rightarrow c^{t+1}$. c_0 is the initial configuration. Cell states of a

configuration c^t are updated simultaneously by the evolution rule as:

$$\varphi(x_{i-r}^t, \dots, x_i^t, \dots, x_{i+r}^t) \rightarrow x_i^{t+1}. \quad (1)$$

where i indicates cell position and r is the radius of neighbourhood μ . Thus, the elementary basic CA class represents a system of order ($k = 2$, $r = 1$) (in Wolfram's nomenclature [61]), the well-known ECA. To represent a specific evolution rule we write name of the rule in a decimal notation, e.g. φ_{R110} .

Conventional cellular automata are memoryless: the new state of a cell depends on the neighbourhood configuration solely at the preceding time step of φ . CA with memory are an extension of ECA in such a way that every cell x_i is allowed to remember its states during some fixed period of its evolution. CA with memory have been proposed originally by Alonso-Sanz [5, 6, 7].

We implement a memory function ϕ as follow:

$$\phi(x_i^{t-\tau}, \dots, x_i^{t-1}, x_i^t) \rightarrow s_i, \quad (2)$$

where $\tau < t$ determines the degree of memory and each cell $s_i \in \Sigma$ is a state function of the series of states of the cell x_i with memory up τ . To execute the evolution we apply the original rule $\varphi(\dots, s_{i-1}^t, s_i^t, s_{i+1}^t, \dots) \rightarrow x_i^{t+1}$. Thus in CA with memory, while the mapping φ remains unaltered, historic memory of all past iterations is retained by featuring each cell as a summary of its past states from ϕ . We can say that cells canalize memory to the map φ [7].

Let us consider the memory function ϕ as a majority memory $\phi_{maj} \rightarrow s_i$, where in case of a tie given by $\Sigma_1 = \Sigma_0$ from ϕ we take the last value x_i . So ϕ_{maj} function represents the classic majority function for three values [47] as follows:

$$(a \wedge b) \vee (b \wedge c) \vee (c \wedge a)$$

that represents the cells $(x_i^{t-\tau}, \dots, x_i^{t-1}, x_i^t)$ and defines a temporal ring before to get the next global configuration c . Of course, this evaluation can be for any number of values. In this way, a number of functional memories may be used such as: minority, parity, alpha, etc. (see [7]).

Evolution rules representation in ECA with memory as given in [30, 31] is as follows:

$$\phi_{CARm:\tau} \quad (3)$$

where CAR is a decimal notation of a particular ECA rule and m is a kind of memory used given with a specific value of τ . Thus for example, the majority memory (maj) incorporated in ECA Rule 30 employing five step of a cell's history ($\tau = 5$) is denoted as $\phi_{R30maj:5}$. The memory is a function of the CA itself, see schematic explanation in Fig. 1.

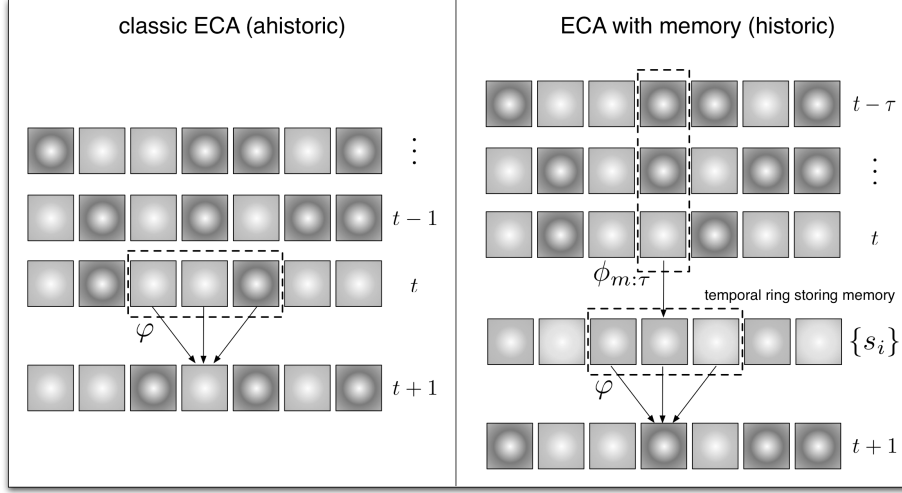


Figure 1: Illustration of cell-state transition in ECA (left) and ECA with memory (right).

3 Toffoli's supercollider

In the late 1970s Fredkin and Toffoli developed a concept of a general-purpose computation based on ballistic interactions between quanta of information that are represented by abstract particles [60]. The Boolean states of logical variables are represented by balls or atoms, which preserve their identity when they collide with each other. They came up with the idea of a billiard-ball model of computation, with underpinning mechanics of elastically colliding balls and mirrors reflecting the balls' trajectories. Later Margolus developed a special class of CA which implement the billiard-ball model. Margolus' partitioned CA exhibited computational universality because they simulated Fredkin gate via collision of soft spheres [36].

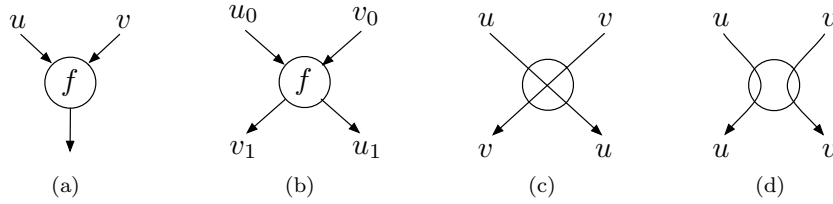


Figure 2: Basic schemes of ballistic collision between localizations representing logical values of the Boolean variables u and v .

The following basic functions with two input arguments u and v can be expressed via collision between two localizations:

1. $f(u, v) = c$, fusion (Fig. 2a)
2. $f(u, v) = u + v$, interaction and subsequent change of state (Fig. 2b)
3. $f_i(u, v) \mapsto (u, v)$ identity, solitonic collision (Fig. 2c);
4. $f_r(u, v) \mapsto (v, u)$ reflection, elastic collision (Fig. 2d);

To map Toffoli's supercollider [60] onto a one-dimensional CA we use the notion of an idealized particle $p \in \Sigma^+$ (without energy and potential energy). The particle p is represented by a binary string of cell states.

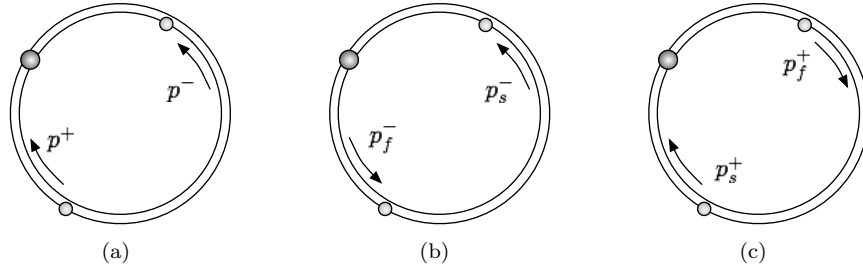


Figure 3: Representation of abstract particles in a one-dimensional CA ring.

Figure 3 shows two typical scenarios where particles p_f and p_s travel in a CA cyclotron. The first scenario (Fig. 3a) shows two particles travelling in opposite directions which then collide. Their collision site is shown by a dark circle in (Fig. 3a). The second scenario demonstrates a typical beam routing where a fast particle p_f eventually catches up with a slow particle p_s at a collision site (Fig. 3b). If the particles collide like solitons [24], then the faster particle p_f simply overtakes the slower particle p_s and continues its motion (Fig. 3c).

Typically, we can find all types of particles manifest in CA gliders, including positive p^+ , negative p^- , and neutral p^0 displacements [44], and also composite particles assembled from elementary localizations. Let us consider the case where a quiescent state is substituted by cells synchronized together as an ether (periodic background). This phenomenon is associated with ECA Rule 110 φ_{R110} .¹ Its evolution space is dominated by a number of particles emerging in various different orders, some of which are really quite complex constructions. Consequently, the number of collisions between particles is increased. Each particle has a period, displacement, velocity, mass, volume, and phase [44, 45].² Figure 4 displays a typical collision between two particles in φ_{R110} . As a result of the collision one particle is split into three different particles (for details please see [37]). The pre-collision positions of particles determines the outcomes of the collision.

¹Rule 110 repository <http://uncomp.uwe.ac.uk/genaro/Rule110.html>

²A full description of particles in Rule 110 is available at <http://uncomp.uwe.ac.uk/genaro/rule110/glidersRule110.html>

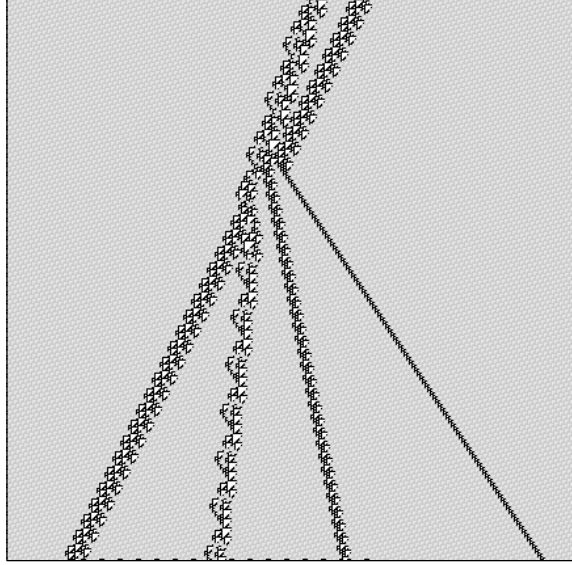


Figure 4: Example of a particle collision in φ_{R110} . Particle $p_{\bar{B}}^-$ collides with particle p_G^- giving rise to three new particles — p_F^- , $p_{D_2}^+$, and $p_{A_3}^+$ — that are generated as a result of the collision.

To represent particles on a given beam routing scheme (see Fig. 3), we do not consider ether configuration in φ_{R110} because this does not affect on collisions. Figure 5 displays a one-dimensional configuration where two particles collide repeatedly and interact as solitonic so that the identities of the particles are preserved in the collisions. A negative particle p_F^- collides and overtakes a neutral particle $p_{C_1}^-$. Figure 5a presents a whole set of cells in state 1 (dark points) where the ether configuration makes it impossible to distinguish the particles: p_F^- and $p_{C_1}^-$. However, we can apply a filter and thereby select particles from their background ether (Fig. 5b).³ Space-time configurations of a cellular automaton exhibiting a collision between particles p_F^- and $p_{C_1}^-$ are shown in Fig. 5c.

Filters selected in CA are a useful tool for understand “hidden” properties of CA. This tool was amply developed by Wuensche in the context of automatic classification of CA [63]. The filters were derived from mechanical computation techniques [22], pattern recognition [57], and analysis of cell-state frequencies [63]. Thus, a filter is a sequence of cells that have a high frequency in the evolution space. Such d -dimensional string repeat periodically, coexisting with any complex structure without altering or disturbing the global dynamics.

³Ring evolution was simulated with DDLab available in <http://www.ddlab.org>.

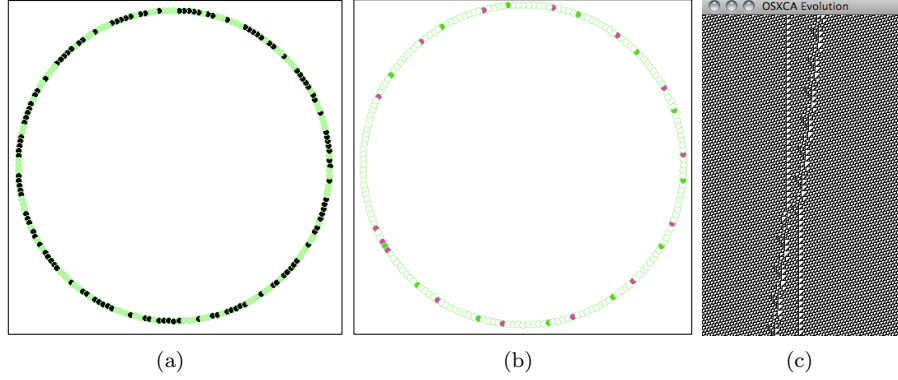


Figure 5: Example of a soliton-type interaction between particles in one-dimensional cellular automaton φ_{R110} : (a)–(b) two steps of beam routing, (c) exact configuration at the time of collision.

4 Ballistic collisions in cellular automata

In this section we analyze examples of ballistic collisions between particles in ECA and ECA with memory. Let us start with ECA with memory governed by the rule $\phi_{R22maj:4}$ [38]. There are only two types of particles in this rule $\mathcal{G}_{\phi_{R22maj:4}} = \{g_L, g_R\}$. Their properties are easy to infer: Both particles have a volume of a perfect square of 11×11 cells, a mass of 35 cells, and they translate 2 cells per 11 time steps (or iterations of CA evolution). The particle g_L has a negative slope with a velocity of $-\frac{2}{11}$ and the particle g_R has a positive slope with a velocity of $\frac{2}{11}$.

The rule $\phi_{R22maj:4}$ supports two types of ballistic collisions: $f_i(u, v)$ and $f_r(u, v)$. Figure 6a shows how a number of soliton interactions can be synchronized as a identity collision $f_i(u, v)$, where both particles can cross their own trajectories. This result can be achieved by selecting a particular phase of each particle as encoded by its initial condition. Different initial conditions lead to different reactions [38]. For example, in Fig. 6b we can see particles undergoing elastic collisions similarly to a lattice gas model [60]. This is a reflection collisions $f_r(u, v)$.

| particle | velocity |
|---------------------|-------------|
| \vec{w} | $2/2 = 1$ |
| \overleftarrow{w} | $-2/2 = -1$ |
| g_o | $0/4 = 0$ |
| g_e | $0/4 = 0$ |
| gun | $0/32 = 0$ |

Table 1: Properties of particles in rule φ_{R54} [32].

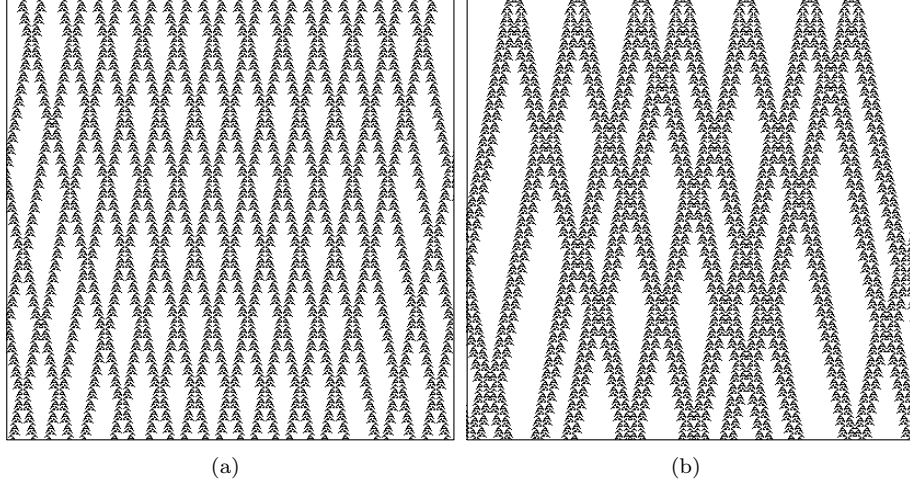


Figure 6: Ballistic collisions in CA $\phi_{R22maj:4}$: (a) identity or soliton collision, and (b) reflections.

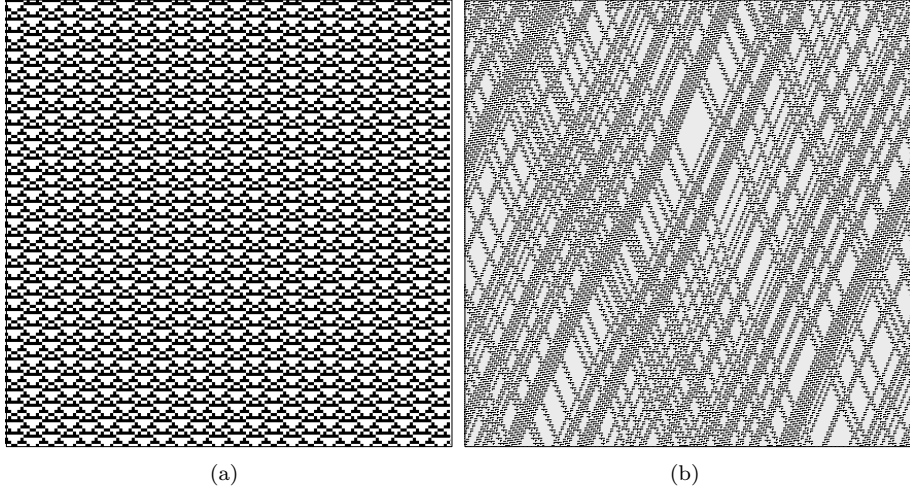


Figure 7: Ballistic collisions simulating the identity relation, or a solitonic reaction: (a) with φ_{R54} and (b) with $\phi_{R9maj:4}$.

Figure 7 presents solitonic reactions (identity ballistic collision) for rules φ_{R54} (memoryless ECA) and $\phi_{R9maj:4}$ (ECA with memory). The rule φ_{R54} has been studied in detail in [11, 22, 32, 33].⁴ In representing particles in φ_{R54} as $\mathcal{G}_{\varphi_{R54}} = \{\vec{w}, \overleftarrow{w}, g_o, g_e, \text{gun}\}$, we follow Boccara's *et al.* notation [11]. Table 1 gives the relation between particles and velocity.⁵ In this way, to produce a required reaction we must code a particular initial condition, where a \vec{w} particle is ready to collide with a \overleftarrow{w} particle and both particles collide with the same phase with a stationary particle g_e (Fig. 7a).

Figure 7(b) shows an ECA with memory which preserves identity ballistic collision starting with any random initial condition. Thus, $\phi_{R9maj:4}$ evolves with two particles $\mathcal{G}_{\phi_{R9maj:4}} = \{\vec{p}, \overleftarrow{p}\}$. The particles' properties are easy to calculate. The \vec{p} particle has a volume of 5×6 cells, a mass of 12 cells, and moves 2 cells in 5 generations (positive slope). While \overleftarrow{p} particle has a volume of 5×3 cells, a mass of 7 cells, and moves 2 cells in 5 generations (negative slope).

As the example above demonstrates, by using ECA and ECA with memory we can experimentally study a variety of ballistic collisions.

5 Beam routings and computations

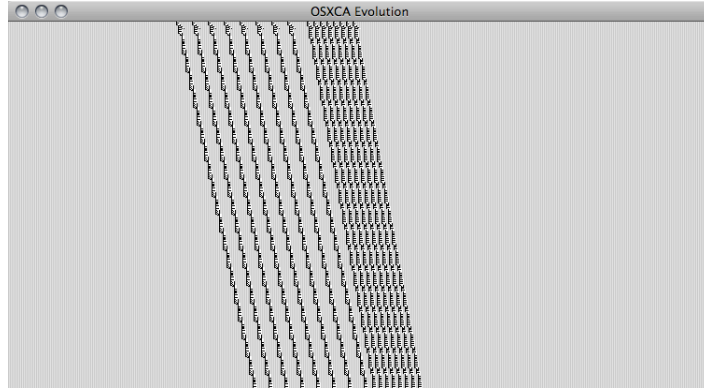
In this section we will exploit beam routing to produce some complex constructions that are based on particle-collisions. An additional effort is necessary to code initial conditions for every particle and recognize the most suitable phase for each particle in order to produce the desired reactions. Also, we will show how the beam routing can be used in design of computing based-collisions.

Let us consider a ECA with memory, rule $\phi_{R30maj:8}$ [30]. We want to implement a simple substitution function **addToHead** working on two strings $w_1 = A_1, \dots, A_n$ and $w_2 = B_1, \dots, B_m$, where $n, m \geq 1$. For example, if $w_1 = AAA$, $w_2 = BBB$ and $w_3 = w_1 w_2$ then **addToHead**($|w_2|$) means produce $w_3 = w_2 w_1$. To implement such a function in $\phi_{R30maj:8}$ every quantum of data is represented by a particle. Particles g_1 and g_2 are coded in order to reproduce a soliton reaction. The codification is not sophisticated. However, a systematic analysis of reactions is required. A periodic gap and one fixed phase between particles is sufficient to reproduce **addToHead** function for any string $A^n B^m$.

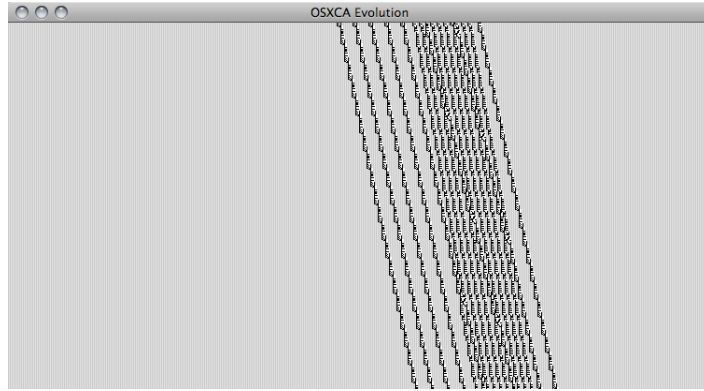
Figure 8 shows the evolution of $\phi_{R30maj:8}$ starting from an initial condition coded by particles representing the string $AAAAAAAAABBBBBBBB$. Using function **addToHead** we produce the final string $BBBBBBBBBAAAAAAAAA$ after 5,000 generations. The first snapshot (Fig. 8a) shows the initial configuration and the first 391 generations, the middle snapshot (Fig. 8b) demonstrates how string w_1 acts on string w_2 while preserving the information (soliton or identity reaction), and the third snapshot (Fig. 8c) shows the final global configuration (i.e. string $w_2 w_1$ processed in parallel by $\phi_{R30maj:8}$).

⁴Rule 54 repository in <http://uncomp.uwe.ac.uk/genaro/Rule54.html>

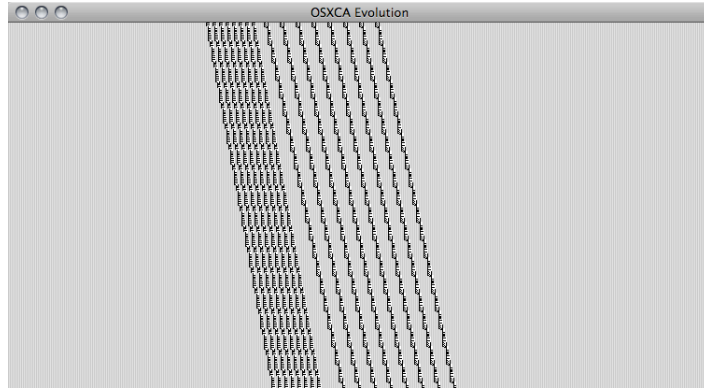
⁵Velocity is calculated as the number of cells a particle displaces in one unit of discrete time.



(a)



(b)



(c)

Figure 8: A simple substitution system processing the word A^8B^8 to B^8A^8 with $\phi_{R30maj:8}$. The final required state is reached at the 5,000th generation by synchronization of multiple soliton reactions.

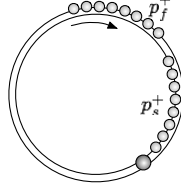


Figure 9: Beam routing performing identity reactions in $\phi_{R30maj:8}$. A cycle realizes its operation, two cycles reinitialize the beam state and the operations can then be repeated.

Therefore a beam routing to represent such an operation in $\phi_{R30maj:8}$ requires two particles (as positrons but one slow and other fast respectively) with the same orientation and one collision contact point on the beam routing. Thus Fig. 9 illustrates how a beam routing is designed to produce such periodic collisions.

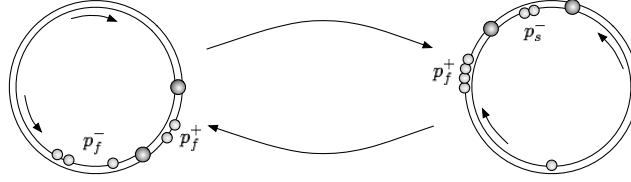


Figure 10: Transition between two beam routing synchronizing multiple reactions. When the first set of collisions are done a new beam routing is defined with other particles, so that when the second set of collisions is done then one returns to the initial condition of the first beam, constructing a meta-glider or mesh in φ_{R110} .

In this way, we can design more complex constructions synchronizing multiple collisions with a diversity of speeds and phases on different particles. Figure 10 displays a more sophisticated beam routing design, connecting two of beams and then creating a new beam routing diagram where edges represent a change of particles and collisions contact point on ECA φ_{R110} . In such a transition, a number of new particles emerge and collide to return to the first beam, thus oscillating between two beam routing forever. To better understand this double beam routing dynamic, consider Fig. 11 where we see multiple collisions between particles (first beam routing):

$$p_A^+, p_A^+ \leftrightarrow p_B^-, p_B^-, p_B^-$$

changing to the set of particles (second beam routing):

$$p_{A^4}^+ \leftrightarrow p_E^+, p_E^+$$

defining two beam routing connected by a transition of collisions as:

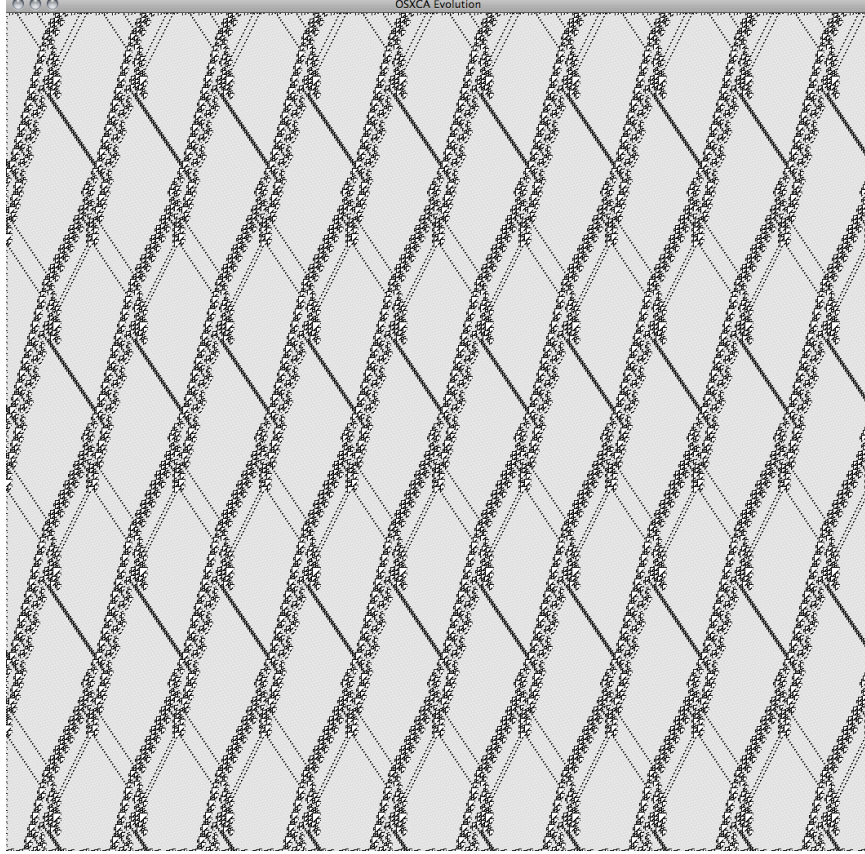


Figure 11: Synchronization of multiple collisions in φ_{R110} on a ring of 1,060 cells in 1,027 generations, starting with 50 particles from its initial condition.

$$(p_A^+, p_A^+ \leftrightarrow p_B^-, p_B^-, p_B^-) \rightarrow (p_{A^4}^+ \leftrightarrow p_E^+, p_E^+), \text{ and} \\ (p_{A^4}^+ \leftrightarrow p_E^+, p_E^+) \rightarrow (p_A^+, p_A^+ \leftrightarrow p_B^-, p_B^-, p_B^-).$$

So we see that a beam routing representation not only helps in designing collisions but also to implement computation.

We now explain the function of a cyclic tag system (CTS) in φ_{R110} in beam routing terms and discuss a number of theoretical implications.

It is well known that ECA φ_{R110} is universal [13, 61]. However, a number of details about its construction, e.g. a cyclic machine, are uncertain [46].⁶ Simplified implementations of universal computation in CTS are provided in [14, 51].

⁶A complete description of CTS working in φ_{R110} is available at <http://uncomp.uwe.ac.uk/genaro/rule110/ctsRule110.html>

| particle | velocity |
|-------------|----------------------------|
| A | $2/3 \approx 0.666666$ |
| B | $-2/4 = -0.5$ |
| B^n | $-6/12 = -0.5$ |
| \hat{B}^n | $-6/12 = -0.5$ |
| C_1 | $0/7 = 0$ |
| C_2 | $0/7 = 0$ |
| C_3 | $0/7 = 0$ |
| D_1 | $2/10 = 0.2$ |
| D_2 | $2/10 = 0.2$ |
| E^n | $-4/15 \approx -0.266666$ |
| \bar{E} | $-8/30 \approx -0.266666$ |
| F | $-4/36 \approx -0.111111$ |
| G^n | $-14/42 \approx -0.333333$ |
| H | $-18/92 \approx -0.195652$ |
| gun | $-20/77 \approx -0.259740$ |

Table 2: Properties of particles in φ_{R110} .

ECA φ_{R110} has a unique complexity due to great number of particles that emerge in the automaton evolution. Table 2 presents a summary of the basic particle properties in φ_{R110} , following Cook's nomenclature. We can appreciate how diverse they are. There are even particles that can expand size forever which increases the number of collisions that we can produce on this ECA.

Initially, a CTS works as a traditional tag system [47], i.e., as a substitution system reading the first symbol on the tape, deleting and putting new symbols. CTS are new machines proposed by Cook [13] as a tool to implement computations in φ_{R110} . CTS are a variant of tag systems: they have the same action of reading a tape in the front and adding characters at the end, nevertheless there are some new features as follows:

1. A CTS needs at least two letters in its alphabet ($\mu > 1$).
2. Only the first character is deleted ($\nu = 1$) and its respective sequence is added.
3. If the machine reads a character zero then the production rule is always null ($0 \rightarrow \epsilon$, where ϵ represents the empty word).
4. There are k sequences from μ^* which are periodically accessed to specify the current production rule when a nonzero character is taken by the system. Therefore the period of each cycle is determinate by k .

This way a cycle determines a partial computation over the tape. No particular halt conditions are specified in the original paper [13], possibly because the halting is a direct consequence of tag systems. Let us see some samples of a CTS working with $\mu = 2$, $k = 3$ and the production rules: $1 \rightarrow 11$, $1 \rightarrow 10$ and $1 \rightarrow \epsilon$.

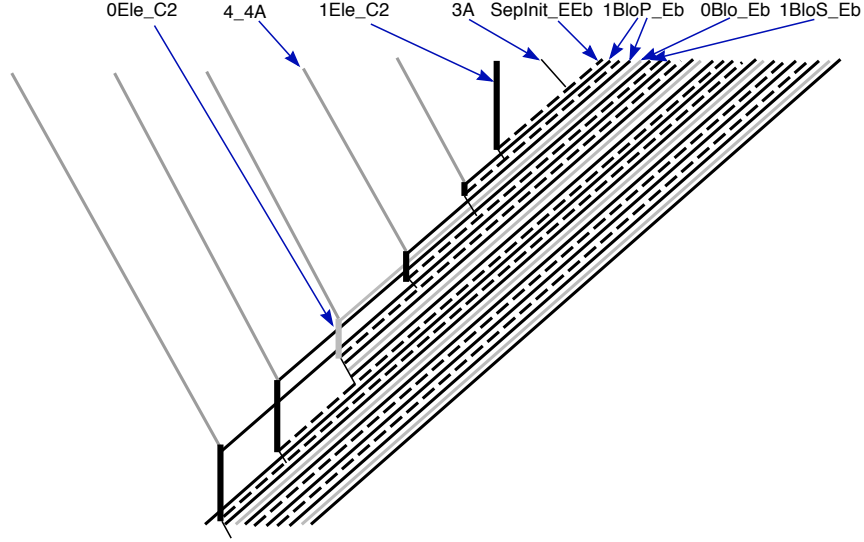


Figure 12: A diagram of a CTS working in φ_{R110} .

Our intention is to represent CTS function in φ_{R110} based beam routing of symbols. We employ transition beam routings to explain critical changes that occur due to collisions and sets of particles that are present on each beam routing at every instant description of the machine.

We start with a description of main stages of encoding packages of particles in their initial condition. This will be an equivalent of beam routing diagram. We will first show how particles and their collisions emulate a CTS in φ_{R110} . We use packages of particles to represent data and operators. We read, transform and delete data in the tape using reactions between the particles. The approach is a delicate one and requires a laborious task of setting initial configurations for the particles in the beams. A diagram of such representation is shown in Fig. 12, particular features of the diagram are explain below.

A construction of the CTS in φ_{R110} can be partitioned, essentially, in three parts:

- First, is the left periodic part, controlled by packages of 4_A^4 particles. This part is static and controls the production of 0's and 1's.
- The second part is the center, determining the initial value in the tape.
- The third one is the right cyclic part, which has the data to process, adding a leader component which specifies data added to or erased from the tape in the evolution space.

In the left part, the four packages of A^4 (Fig. 13c) particles must be carefully explained because although they are static, their phases change periodically. The important point is to implement these components in defining both

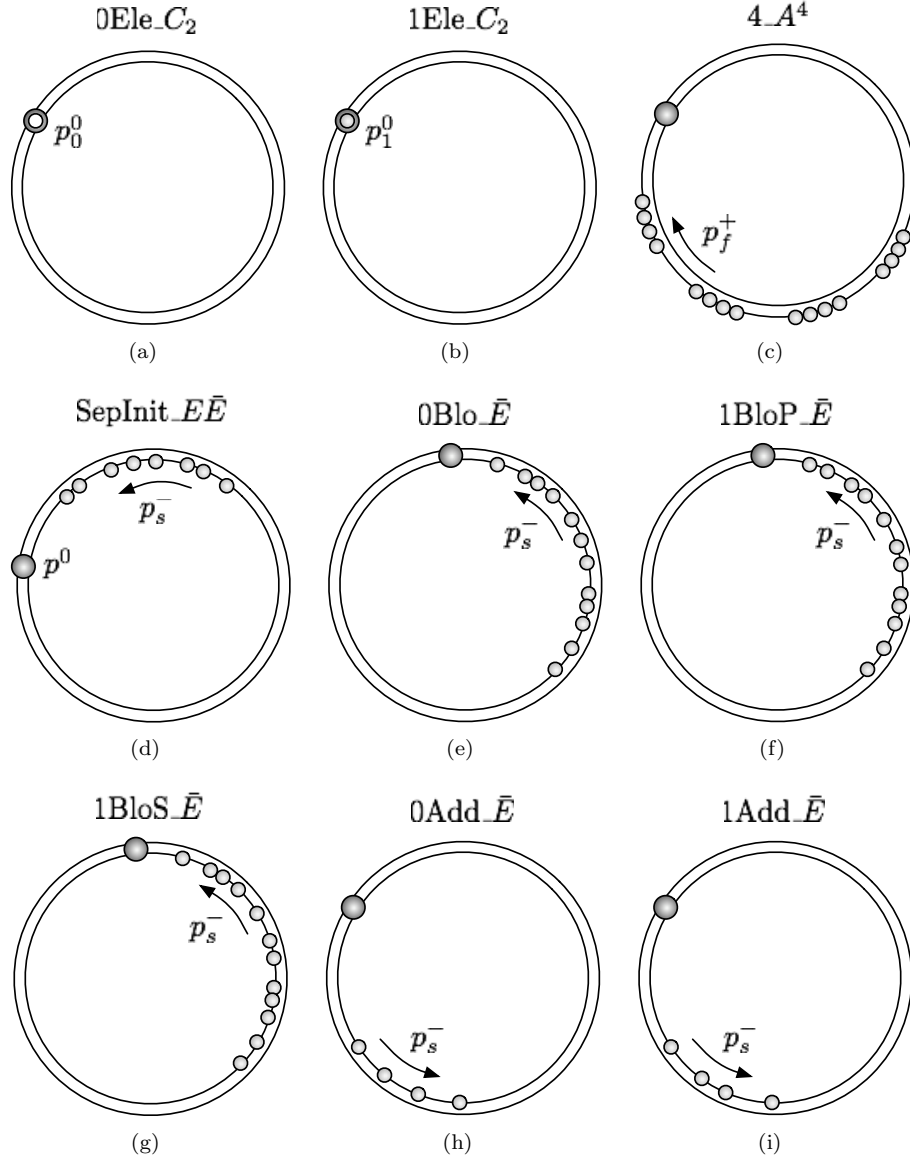


Figure 13: Beam routing codification representing package of particles which reproduces a CTS in φ_{R110} .

distances and phases, because a distinct phase or a distance induces an undesirable reaction; in fact all components obey this restriction because every glider of each component must be correctly aligned.

The central part is constituted by an initial 1 on the tape represented by a package of four C_2 particles. This way, an element 1Ele_{C_2} (Fig. 13b) represents a 1 and the element 0Ele_{C_2} (Fig. 13a) represents a 0 in the tape.

The element $0\text{Blo}_{\bar{E}}$ is formed by $12\bar{E}$ particles. For the $1\text{Blo}_{\bar{E}}$ element, Cook establishes the existence of two components to represent 1's: $1\text{BloP}_{\bar{E}}$ (Fig. 13f) named *primary* and $1\text{BloS}_{\bar{E}}$ (Fig. 13g) named *standard*. In essence both blocks produce the same element $1\text{Add}_{\bar{E}}$, although coming from different intervals. The reason to use both blocks is that φ_{R110} is not symmetric; thus, if we use only one element then although we would obtain a good production in the first collision, generating an element $1\text{Add}_{\bar{E}}$, when the second package of 4_{A^4} particles finds the second block, the system is completely destroyed.

A leader element $\text{SepInit}_{E\bar{E}}$ (Fig. 13d) is important in order to separate packages of data and determine their incorporation into of the tape.

The elements $1\text{Add}_{\bar{E}}$ (Fig. 13i) and $0\text{Add}_{\bar{E}}$ (Fig. 13h) are produced by two previous different packages of data. An element $1\text{Add}_{\bar{E}}$ must be generated by a block $1\text{BloP}_{\bar{E}}$ or by $1\text{BloS}_{\bar{E}}$. This way, both elements can produce the same element. While an element $0\text{Add}_{\bar{E}}$ is generated by a block $0\text{Blo}_{\bar{E}}$ (Fig. 13e).

Finally, in terms of periodic phases, this CTS can be written as follow:

left: $\{649e-4_{A^4}(\text{F}_{-i})\}^*$, for $1 \leq i \leq 3$ in sequential order

center: $246e-1\text{Ele}_{C_2}(\text{A}, \text{f}_{1-1})-e-A^3(\text{f}_{1-1})$

right: $\{\text{SepInit}_{E\bar{E}}(\#, \text{f}_{i-1})-1\text{BloP}_{\bar{E}}(\#, \text{f}_{i-1})-\text{SepInit}_{E\bar{E}}(\#, \text{f}_{i-1})-1\text{BloP}_{\bar{E}}(\#, \text{f}_{i-1})-0\text{Blo}_{\bar{E}}(\#, \text{f}_{i-1})-1\text{BloS}_{\bar{E}}(\#, \text{f}_{i-1})\}^*$ (where $1 \leq i \leq 4$ and $\#$ represents a particular phase).

For a complete and full description of such reproduction by phases f_{i-1} , please see [46].

To get a CTS emulation in φ_{R110} by beam routings, we will use connections between beam routing, as we have proposed in Fig 10.

Transitions between beam routings connect each set of collisions to enter to the next beam routing diagram.

We need some fine details to get a CTS operation by beam routings. Figure 14 shows the general diagram to reproduce a CTS by particle collisions in φ_{R110} with beam routing transitions. While Fig. 13 describes each component to code particles in φ_{R110} , Fig. 14 display how to connect such components and control the transition of collisions.

Some notes are necessary to better understand this schematic diagram. For the components 1Ele_{C_2} and 0Ele_{C_2} they are compressed only for one dark circle (that represents the point of collisions). Both elements are constituted for four C_2 particles in different distances, although they have a static position p^0 that can be confined to this dark circle, as p_1^0 and p_0^0 respectively.

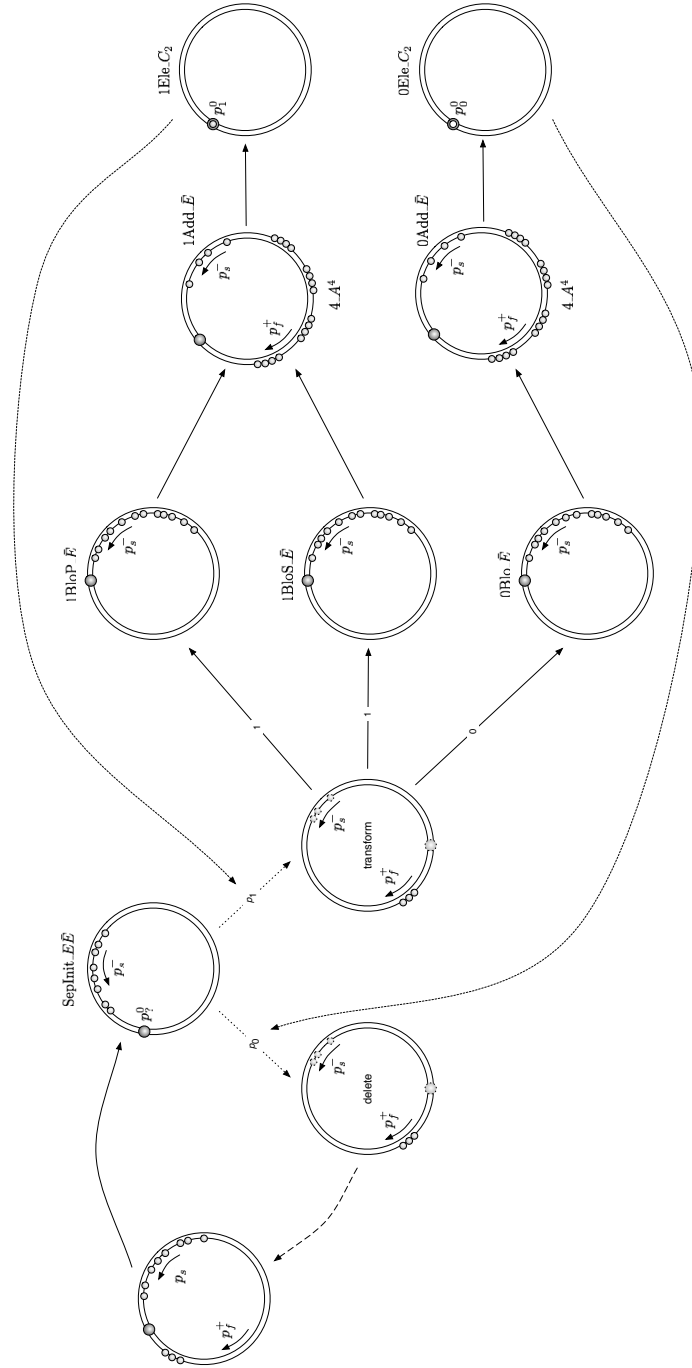


Figure 14: Beam routing machine transitions to simulate CTS in φ_{R110} .

When a leader component (SepInit_ $E\bar{E}$) is transformed, given previous binary value on the tape, it collides with p_7^0 component, i.e. a p_1^0 or p_0^0 element. If p_7^0 is 0, then a cascade of collisions start to *delete* all components that come with three particles successively and change this reaction to reach a new leader component and return to the beginning. But, if p_7^0 is 1 then a cascade of *transformations* dominated by additional particles p^0 starts, in order to reach the next leader component. Here, we have more variants because pre-transformed package of particles is encoded to binary values that is then included on the machine tape. If a block of particles is 1BloP_ \bar{E} or 1BloS_ \bar{E} this means that such a component will be transformed to one 1Add_ \bar{E} element. But, if a block of particles is 0Blo_ \bar{E} , then such a component will be transformed to 0Add_ \bar{E} element. At this stage, when both components are prepared then a binary value is introduced on the tape, a 1Add_ \bar{E} element stores a 1 (1Ele_ C_2), and a 0Add_ \bar{E} element stores a 0 (0Ele_ C_2), which eventually is deleted for the next leader component and starts a new cycle in the CTS machine.

The relevant point here is how to control and code particles by beam routings. That will offer a better chance to perform a computation with this architecture before implementing the complicated and laborious task of setting up initial condition. Of course, such an idea can be carried out in any CA handling signals, waves, particles, gliders or mobile self-localizations [1, 2, 3, 9, 12, 24, 26, 27, 29, 32, 34, 35, 36, 43, 48, 49, 52, 56, 23].

The package particles codification for a CTS in [61, 46] is represented for the expression (s1s101)⁺ while that in [13] is represented for the expression (s111s0)⁺. Finally in all cases, this CTS beam routing may simulate such operations.

6 Summary

We advanced a concept of symbol super collider [60] and implemented it in one-dimensional cellular automata. Two types of automaton rings were considered — elementary cellular automata (binary cell state, ternary neighbourhoods) and elementary cellular automata with memory. We demonstrated that, for certain rules of cell-state transition functions, the automata exhibit a wide range of particles (mobile self-localizations or gliders) in their evolution. High degree of morphological diversity and richness of collision dynamics of the particles in memory-enriched cellular automata make them particularly promising objects for constructing computational schemes.

In present paper we advanced over twenty years old studies in identifying and classifying gliders and localisations in CA. There is a number of approaches varying in their degrees of efficiency and discovery potentials. Thus, Wuensche successfully classified CA evolution rules with his Z parameter, and provided power structures of a substantial number of glider-generating rules [63]. Other studies focus on specification of sets of strings as patterns and gliders. For example, Redeker developed a compact algebraic representation of 1D CA evolution, known as *flexible time*. Suing his approach one can construct specific periodic

structures [54]. Evolution programming techniques are also proved to be very successful, e.g. in situations when a genetic algorithm undertakes a search for gliders and patterns on fixed small arrays of bits [55, 62]. As well, Freire *et. al* developed an approach to select specific sets of vectors, initial configurations, that determine development of gliders and localised patterns [18, 19]. Finally, the de Bruijn diagrams is a time-tested and reliable tool to classify sets of periodic structures or gliders using exhaustive approach [40, 41]. Said that in present paper we did not aim to deal with identification of mobile patterns but rather consider glider sets in collision-based computing perspective. Concretely, we only aimed to develop the idea of cyclotrons as an abstract model of super-colliders [60].

We proved experimentally that the dynamics of these particles support all basic types of ballistic collisions and consequently that the cellular automata cyclotrons can be used to implement certain schemes and operations of collision-based computing. We designed schemes with several beams of particles and provided a means of beam routing and programming interactions between particles. To demonstrate the high degree of computational universality of cellular automaton ‘cyclotrons’ we took into account that each particle is essentially a finite-size binary string travelling along a one-dimensional lattice ‘cyclotron.’ When a string with lower velocity overtaken by a string with higher velocity the content of one or both strings can be modified. Thus, for carefully selected initial configuration of particles and types of particles on a ring, the cellular automaton cyclotron imitates a cyclic tag systems, and thus is computationally universal.

What are advantages of the proposed approach? In cellular-automaton models particles can run along the rings indefinitely. With suitable beam routing schemes some particles (the results of computation or the products of reactions) can be removed from the system and new particles can be added. The cellular-automaton super collider is a universal computing devices based on interactions between particles due to the particles’ different speeds. This give our constructs an enormous advantage when compared to more “traditional” constructs, where particles collide only due to different orientations of velocity vectors. Also most existing unconventional designs of universal computing devices have certain issues of boundary conditions, e.g. infinite versus finite, the cellular-automaton super-colliders by using periodic boundary conditions removes the effect of any boundary being equivalent to an infinite periodic system.

The theoretical models of cellular-automaton super colliders open new perspectives in laboratory implementations of collision based computing devices. The opportunities are virtually endless. Hundreds of chemical, physical and biological systems exhibit travelling localizations in their dynamics. Examples include co-aligned dipole groups in tubulin microtubules (e.g. [53, 39], kinks, breathers and solitons in molecular chains and polymers (e.g. [16], phasons in quasi-crystals [28], kinks in ferromagnets [15], dissipative solitons in gas-discharge systems [8], localizations of electron density in monolayers of aromatic molecules [10], wave-fragments in excitable chemical media [4]. We envisage that molecular chains and polymers would be the best candidates for experimental

laboratory realisation of symbol super colliders. This will be a subject of further studies.

Acknowledgement

Genaro J. Martínez thanks to support given by DGAPA-UNAM and EPSRC grant EP/F054343/1.

References

- [1] Adamatzky, Andrew, *Computing in Nonlinear Media and Automata Collectives*, Institute of Physics Publishing, Bristol and Philadelphia 2001.
- [2] Adamatzky, Andrew (Ed.) *Collision-Based Computing*, Springer 2002.
- [3] Adamatzky, Andrew, “New media for collision-based computing,” In [2], 411–442, 2002.
- [4] Adamatzky, Andrew, Costello, Ben L., and Asai, Tetsuya, *Reaction-Diffusion Computers*, Elsevier 2005.
- [5] Alonso-Sanz, R. and Martin, M., “Elementary CA with memory,” *Complex Systems* **14** 99–126, 2003.
- [6] Alonso-Sanz, R., “Elementary rules with elementary memory rules: the case of linear rules,” *Journal of Cellular Automata* **1** 71–87, 2006.
- [7] Alonso-Sanz, Ramon, *Cellular Automata with Memory*, Old City Publishing 2009.
- [8] Astrov Y., “Gas-discharge planar semiconductor structures as devices for un-conventional computing,” *Int. J. Unconventional Computing*, in press.
- [9] Awazu, Akinori, “Cellular automaton rule 184++C. A simple model for the complex dynamics of various particles flow,” *Physics Letters A* **261**(5-6) 309–315, 1999.
- [10] Bandyopadhyay A., Pati R., Sahu S., Peper F., and Fujita D., “Massively parallel computing on an organic molecular layer,” *Nature Physics* **6** 369–375, 2010.
- [11] Boccara, Nino, Nasser, J., and Roger, M. “Particle like structures and their interactions in spatio-temporal patterns generated by one-dimensional deterministic cellular automaton rules,” *Physical Review A* **44**(2) 866–875, 1991.
- [12] Chopard, Bastien and Droz, Michel, *Cellular Automata Modeling of Physical Systems*, Collection Aléa Saclay, Cambridge University Press 1998.

- [13] Cook, Matthew, “Universality in Elementary Cellular Automata,” *Complex Systems* **15(1)** 1–40, 2004.
- [14] Cook, Matthew, “A Concrete View of Rule 110 Computation,” In *The Complexity of Simple Programs* (T. Neary, D. Woods, A. K. Seda, and N. Murphy (Eds.)), 31–55, 2008.
- [15] R. Coldea, D. A. Tennant, E. M. Wheeler, E. Wawrzynska, D. Prabhakaran, M. Telling, K. Habicht, P. Smeibidl, and K. Kiefer, “Quantum criticality in an Ising chain: Experimental evidence for emergent E8 symmetry,” *Science* **327** 177–180, 2010.
- [16] Davydov A. S., *Solitons in Molecular Systems*, Springer 1990.
- [17] Fredkin, Edward and Toffoli, Tommaso, “Design Principles for Achieving High-Performance Submicron Digital Technologies,” In [2], 27–46, 2001.
- [18] Freire, Joana G., Brison, Owen J., and Gallas, Jason A.C. (2010) “Spatial updating, spatial transients, and regularities of a complex automaton with nonperiodic architecture,” *Chaos* **17** 026113.
- [19] Freire, Joana G., Brison, Owen J., and Gallas, Jason A.C. (2010) “Complete sets of initial vectors for pattern growth with elementary cellular automata,” *Computer Physics Communications* **181** 750–755.
- [20] Michael J. Fuerstman, Pascal Deschatelets, Ravi Kane, Alexander Schwartz, Paul J. A. Kenis, John M. Deutch, and George M. Whitesides, “Solving mazes using microfluidic networks,” *Langmuir* **19** 4714–4722, 2003.
- [21] Górecki J., Yoshikawa K., and Igarashi Y., “On chemical reactors that can count,” *J. Phys. Chem. A* **107** 1664–1669, 2003.
- [22] Hanson, James E. and Crutchfield, James P., “Computational Mechanics of Cellular Automata: An Example,” *Physics D* **103** 169–189, 1997.
- [23] Hey, Anthony J. G., *Feynman and computation: exploring the limits of computers*, Perseus Books 1998.
- [24] Jakubowski, Mariusz H., Steiglitz, Kenneth, and Squier, Richard, “Computing with Solitons: A Review and Prospectus,” *Multiple-Valued Logic* **6(5-6)** 439–462, 2001.
- [25] Katz E. and Privman V., “Enzyme-based logic systems for information processing,” *Chem. Soc. Rev.* **39** 1835–1857, 2010. [arXiv:0910.0270](#)
- [26] Kohyama, Tamotsu, “Cellular automata with particle conservation,” *Progress of Theoretical Physics* **81(1)** 47–59, 1989.
- [27] Kůrka, Petr, “Cellular automata with vanishing particles,” *Fundamenta Informaticae* **58(3-4)** 203–221, 2003.

- [28] Lipp G., Engel M., Sonntag S., and Trebin H. R., Phason, “Dynamics in One-Dimensional Lattices,” *arXiv:1002.1918v1*, 2010.
- [29] Lizier, Joseph T., Prokopenko, Mikhail, and Zomaya, Albert Y., “Information transfer by particles in cellular automata,” *Lecture Notes in Computer Science* **4828** 49–60, 2007.
- [30] Martínez, Genaro J., Adamatzky, Andrew, Alonso-Sanz, Ramon, and Seck-Tuoh-Mora, J. C., “Complex dynamic emerging in Rule 30 with majority memory,” *Complex Systems* **18(3)** 345–365, 2010.
- [31] Martínez, Genaro J., Adamatzky, Andrew, Seck-Tuoh-Mora, J. C., and Alonso-Sanz, Ramon, “How to make dull cellular automata complex by adding memory: Rule 126 case study,” *Complexity* **15(6)** 34–49, 2010.
- [32] Martínez, Genaro J., Adamatzky, Andrew, and McIntosh, Harold V., “Phenomenology of glider collisions in cellular automaton Rule 54 and associated logical gates,” *Chaos, Solitons and Fractals* **28** 100–111, 2006.
- [33] Martínez, Genaro J., Adamatzky, Andrew, and McIntosh, Harold V., “On the representation of gliders in Rule 54 by de Bruijn and cycle diagrams,” *Lecture Notes in Computer Science* **5191** 83–91, 2008.
- [34] Margolus, Norman, “Physics-like models of computation,” *Physica D* **10(1-2)** 81–95, 1984.
- [35] Margolus, Norman, “Crystalline computation,” In [23] 267–305, 1999.
- [36] Margolus, Norman, “Universal Cellular Automata Based on the Collisions of Soft Spheres,” In [1] 231–260, 2003.
- [37] Martínez, Genaro J. and McIntosh, Harold V., “ATLAS: Collisions of gliders like phases of ether in rule 110,” http://uncomp.uwe.ac.uk/genaro/Papers/Papers_on_CA.html, 2001.
- [38] Martínez, Genaro J., Morita, Kenichi, Adamatzky, Andrew, and Margenstern, Maurice, “Logic Gates in Elementary Cellular Automata with Memory,” *in elaboration*.
- [39] Mershin A., Kolomenski A. A., Schuessler H. A., Nanopoulos D. V., “Tubulin dipole moment, dielectric constant and quantum behavior: computer simulations, experimental results and suggestions,” *BioSystems* **77** 73–85, 2004. *arXiv:physics/0402053v1*
- [40] McIntosh, Harold V. (1990) “Wolfram’s Class IV and a Good Life,” *Physica D* **45** 105–121.
- [41] McIntosh, Harold V. (2009) *One Dimensional Cellular Automata*, Luniver Press.

- [42] Mills, Jonathan W., “The nature of the Extended Analog Computer,” *Physica D* **237** 1235–1256, 2008.
- [43] Morita, Kenichi, Margenstern, Maurice, and Imai, Katsunobu, “Universality of reversible hexagonal cellular automata,” *Theoret. Informatics Appl.* **33** 535–550, 1999.
- [44] Martínez, Genaro J., McIntosh, Harold V., and Seck-Tuoh-Mora, J. C., “Gliders in Rule 110,” *Int. J. of Unconventional Computing* **2(1)** 1–49, 2006.
- [45] Martínez, Genaro J., McIntosh, Harold V., Seck-Tuoh-Mora, Juan C., and Chapa-Vergara, Sergio V., “Determining a regular language by glider-based structures called *phases f_i-1* in Rule 110,” *Journal of Cellular Automata* **3(3)** 231–270, 2008.
- [46] Martínez, Genaro J., McIntosh, Harold V., Seck-Tuoh-Mora, Juan C., and Chapa-Vergara, Sergio V., “Reproducing the cyclic tag system developed by Matthew Cook with Rule 110 using the phases f_1-1 ,” *Journal of Cellular Automata* **6(2-3)** 121–161, 2011.
- [47] Minsky, Marvin, *Computation: Finite and Infinite Machines*, Prentice Hall 1967.
- [48] Moreira, Andrés, Boccara, Nino, and Goles, Eric, “On conservative and monotone one-dimensional cellular automata and their particle representation,” *Theoretical Computer Science* **352(2)** 285–316, 2004.
- [49] Morita, Kenichi, “Reversible computing and cellular automata—A survey”, *Theoretical Computer Science* **395** 101–131, 2008.
- [50] Nakagaki T., Yamada H., and Toth A., “Maze-solving by an amoeboid organism,” *Nature* **407** 470, 2000.
- [51] Neary, Turlough and Woods, Damien, “P-completeness of cellular automaton Rule 110,” *Lecture Notes in Computer Science* **4051** 132–143, 2006.
- [52] Pivato, Marcus, “Defect particle kinematics in one-dimensional cellular automata,” *Theoretical Computer Science* **377** 205–228, 2007.
- [53] Rasmussen, S., Karampurwala, H., Vaidyanath, R., Jensen, K.S., and Hameroff, S., “Computational connectionism within neurons: A model of cytoskeletal automata subserving neural networks,” *Physica D* **42** 428–449, 1990.
- [54] Redeker, Markus (2010) “Flexible Time and the Evolution of One-Dimensional Cellular Automata,” *Journal of Cellular Automata* **5(4-5)** 273–287.

- [55] Sapin, E., Bailleux, O., Chabrier, J. J., and Collet, P. (2004) “A New Universal Cellular Automaton Discovered by Evolutionary Algorithms,” *Lecture Notes in Computer Science* **3102** 175–187.
- [56] Lun Shi, Fangyue Chen, and Weifeng Ji, “Gliders, Collisions and Chaos of Cellular Automata Rule 62,” *International Workshop on Chaos-Fractals Theories and Applications*, 221–225, 2009.
- [57] Shalizi, Cosma R., Haslinger, Robert, Rouquier, Jean-Baptiste, Klinkner, Kristina L., and Moore, Christopher, “Automatic filters for the detection of coherent structure in spatiotemporal systems”, *Physical Review E* **73(3)** 036104, 2005.
- [58] Stojanovic, M. N. and Stefanovic D. (2003) “Deoxyribozyme-based half-adder,” *J. Am. Chem. Soc.* **125** 6637.
- [59] Toffoli, Tommaso, “Non-Conventional Computers”, In *Encyclopedia of Electrical and Electronics Engineering* **14** (John Webster Ed.), Wiley & Sons, 455–471, 1998.
- [60] Toffoli, Tommaso, “Symbol Super Colliders,” In [2], 1–22, 2002.
- [61] Wolfram, Stephen, *A New Kind of Science*, Wolfram Media, Inc., Champaign, Illinois 2002.
- [62] Wolz, D. and de Oliveira, P.P.B. (2008) “Very effective evolutionary techniques for searching cellular automata rule spaces,” *Journal of Cellular Automata* **3(4)** 289–312.
- [63] Wuensche, Andrew, “Classifying Cellular Automata Automatically,” *Complexity* **4(3)** 47–66, 1999.